AMENDMENTS TO THE CLAIMS

Please amend the claims as follows.

1. (Currently Amended) A method for storing a data block, comprising:

storing the data block in a storage pool;

obtaining a data block location associated with the data block;

determining a checksum function for the data block;

calculating a data block checksum using the checksum function for the data block; ~~and~~

storing a first indirect block in the storage pool, wherein the first indirect block comprises the data block location, the data block checksum, and a checksum function ID corresponding to the checksum function for the data block;[[.]]

determining a checksum function for the first indirect block, wherein the checksum function for the first indirect block is associated with a checksum function ID corresponding to the checksum function for the first indirect block;

calculating a first indirect block checksum using the checksum function for the first indirect block;

obtaining a first indirect block location associated with the first indirect block; and

storing a second indirect block in the storage pool, wherein the second indirect block comprises the first indirect block location, the first indirect block checksum, and the checksum function ID corresponding to the checksum function for the first indirect block.

2. (Canceled)

3. (Original) The method of claim 1, further comprising:

assembling the first indirect block, wherein assembling the first indirect block comprises populating a block pointer in the first indirect block.

4. (Original) The method of claim 3, wherein populating the block pointer comprises:

storing the data block checksum in a checksum field in the block pointer,

storing the checksum function ID for the data block in a checksum function ID field in the block pointer; and

storing the data block location in the block pointer, wherein storing the data block location comprises storing a metaslab ID and offset.

5. (Original) The method of claim 4, further comprising:

storing a birth value in a birth field in the block pointer.

6. (Original) The method of claim 3, wherein the first indirect block is assembled using a data management unit.

7. (Original) The method of claim 1, wherein the storage pool comprises at least one storage device.

8. (Original) The method of claim 1, wherein the storage pool is divided into a plurality of metaslabs.

9. (Original) The method of claim 8, wherein each of the plurality of metaslabs is associated with a metaslab ID.

10. (Original) The method of claim 8, wherein the data block location comprises the metaslab ID and an offset.

11. (Original) The method of claim 1, wherein storing the data block comprises using a storage pool allocator.

12. (Currently Amended) A method for storing a first data block and a second data block, comprising:

storing the first data block and the second data block in a storage pool;

obtaining a first data block location and a second data block location;

selecting a first checksum function and a second checksum function from a plurality of checksum functions, wherein the first checksum function is associated with a first checksum function ID and the second checksum function is associated with a second checksum function ID;

calculating a first data block checksum for the first data block using [[a]] the first checksum function, wherein the first checksum function is associated with [[a]] the first checksum function ID;

calculating a second data block checksum for the second data block using [[a]] the second checksum function, wherein the second checksum function is associated with [[a]] the second checksum function ID; and

storing an array of block pointers in an indirect block, wherein the array of block pointers comprises:

a first block pointer comprising the first data block location, the first data block checksum, and the first checksum function ID, and

a second block pointer comprising the second data block location, the second data block checksum, and the second checksum function ID.

13. (Original) The method of claim 12, wherein the indirect block is assembled using a data management unit.

14. (Original) The method of claim 12, wherein the indirect block is stored using a storage pool allocator.

15. (Currently Amended) A method for retrieving data in a data block, comprising:

obtaining [[an]] a first indirect block comprising a first stored checksum, a first stored checksum function ID, and a second ~~data~~ indirect block location;

obtaining a second indirect block using the second indirect block location, wherein the second indirect block comprises a second stored checksum, a second stored checksum function ID, and a data block location;

obtaining the data block using the data block location;

calculating the checksum for the data block using a checksum function corresponding to the second stored checksum ID to obtain a calculated checksum;

retrieving the data from the data block, if the second stored checksum equals the calculated checksum; and

performing an appropriate action, if the second stored checksum is not equal to the calculated checksum.

16. (Currently Amended) The method of claim 15, wherein the indirect block is assembled using a data management unit.

17. (Original) The method of claim 15, wherein the calculated checksum is calculated using a storage pool allocator.

18. (Currently Amended) A method for storing and retrieving a data block, comprising:

storing the data block in a storage pool;

obtaining a data block location associated with the data block;

determining a checksum function for the data block;

calculating a data block checksum using the checksum function for the data block;

storing [[an]] a first indirect block in the storage pool, wherein the first indirect block comprises the data block location, the data block checksum, and a data block checksum function ID corresponding to the checksum function for the data block;

determining a checksum function for the first indirect block, wherein the checksum function for the first indirect block is associated with a first indirect block checksum function ID corresponding to the checksum function for the first indirect block;

calculating a first indirect block checksum using the checksum function for the first indirect block;

obtaining a first indirect block location; and

storing a second indirect block in the storage pool, wherein the second indirect block comprises the first indirect block location, the first indirect block checksum, and the first indirect block checksum function ID;

obtaining the first indirect block comprising the data block checksum, the data block checksum function ID, and the data block location;

obtaining the data block using the data block location;

calculating the checksum for the data block using [[a]] the checksum function for the data block corresponding to the data block checksum function ID to obtain a calculated checksum;

retrieving the data from the data block, if the data block checksum equals the calculated checksum; and

performing an appropriate action, if the data block checksum is not equal to the calculated checksum.

19. (Currently Amended) A system for storing a data block, comprising:

a storage pool comprising:

the data block; ~~and the~~

a first indirect block, wherein the first indirect block comprises a data block checksum, a data block checksum function ID, and a data block location, wherein the data block checksum is calculated using a data block checksum function, wherein the data block checksum function corresponds to the data block checksum function ID; [[and]]

a second indirect block, comprising a first indirect block checksum, a first indirect block checksum function ID, and a first indirect block location, wherein the first indirect block checksum is calculated using a first indirect block checksum function, wherein the first indirect checksum function corresponds to the first indirect block checksum function ID;

wherein the data block checksum function ID and the first indirect block checksum function ID correspond to checksum functions selected from a plurality of checksum functions implemented by the system; and

a storage pool allocator configured to store [[:]] the data block, [[and]] the first indirect block, and the second indirect block in the storage pool~~, and~~

~~a plurality of checksum functions, wherein each of the plurality of checksum functions is associated with a checksum function ID~~.

20. (Canceled)

21. (Original) The system of claim of claim 19, further comprising:

a data management unit configured to assemble the first indirect block and request the storage pool allocator store the first indirect block.

22. (Original) The system of claim 19, wherein the storage pool comprises at least one storage disk.

23. (Original) The system of claim 19, wherein the storage pool is divided into a plurality of metaslabs.

24. (Original) The system of claim 23, wherein each of the plurality of metaslabs is associated with a metaslab ID.

25. (Original) The system of claim 24, wherein the data block location comprises the metaslab ID and an offset.

26. (Currently Amended) A computer system for storing a data block, comprising:

a processor;

a memory;

a storage device; and

software instructions stored in the memory for enabling the computer system under control of the processor, to:

store the data block in a storage pool;

obtain a data block location;

determine a checksum function for the data block;

calculate a data block checksum using the checksum function for the data block; [[and]]

store a first indirect block in the storage pool, wherein the first indirect block comprises the data block location, the data block checksum, and a <u>data block</u> checksum function ID corresponding to the checksum function for the data block[[.]];

<u>determine a checksum function for the first indirect block, wherein the checksum function for the first indirect block is associated with a first indirect block</u>

checksum function ID corresponding to the checksum function for the first indirect block;

calculate a first indirect block checksum using the checksum function for the first indirect block;

obtain a first indirect block location; and

store a second indirect block in the storage pool, wherein the second indirect block comprises the first indirect block location, the first indirect block checksum, and the first indirect block checksum function ID.

27. (Currently Amended) A network system having a plurality of nodes, comprising:

a storage pool comprising:

[[the]] a data block; ~~and the~~

a first indirect block, wherein the first indirect block comprises [[a]] the data block checksum, [[a]] the data block checksum function ID, and [[a]] the data block location, wherein the data block checksum is calculated using the data block checksum function, wherein the data block checksum function corresponds to the data block checksum function ID;

a second indirect block, wherein the second indirect block comprises the first indirect block checksum, the first indirect block checksum function ID, and the first indirect block location, wherein the first indirect block checksum is calculated using the first indirect block checksum function, wherein the first indirect block checksum function corresponds to the first indirect block checksum function ID;

wherein the data block checksum function ID and the first indirect block checksum function ID correspond to checksum functions selected from a plurality of checksum functions implemented by the system; and

a storage pool allocator configured to store [[:]] the data block, [[and]] the first indirect block, and the second indirect block in the storage pool;~~, and~~

~~a plurality of checksum functions, wherein each of the plurality of checksum functions is associated with a checksum function ID,~~

wherein the storage pool is located on any one of the plurality of nodes[[,]]; and

wherein the storage pool allocator is located on any one of the plurality of nodes.